Amendments to the Specification:

Please replace paragraph beginning at page 10, line 21, which has been previously amended by a Preliminary Amendment, with the following amended paragraph:

Referring now to Figure 2, an architecture for implementation of an embodiment of an ICE system of the present invention is illustrated as system 200. In system 200, a host computer 210 (e.g., personal computer based on a PENTIUM® class microprocessor) is connected (e.g., using a production PC interface 214 such as a parallel printer port connection, a universal serial ~~port~~ bus (USB) connection, etc.) with a base station 218. The host computer 210 generally operates to run an ICE computer program to control the emulation process and further operates in the capacity of a logic analyzer to permit a user to view information provided from the base station 218 for use in analyzing and debugging a device under test or development. According to an embodiment of the present invention microcontroller 232, mounted on a pod, comprises file registers 236, SRAM 237, CPU register 238, memory 240, and a program counter 239.

Please replace paragraph beginning at page 11, line 6, which has been previously amended by an Amendment mailed 1/10/2006 in response to an Office Action mailed 11/21/2005, with the following amended paragraph:

The base station 218 is based upon a general purpose programmable hardware device such as a gate array configured to function as a functionally equivalent "virtual microcontroller" 220. This is accomplished using an

associated integral memory 222 which stores program instructions, data, and other associated information. Base station 218 comprises ~~of~~ file register 221, SRAM 225, CPU register 234 and program counter 233. The base station 218 is configured as an emulator of the internal microprocessor portion of the microcontroller portion of the microcontroller 232. In preferred embodiments, a field programmable gate array FPGA (or other programmable logic device) is configured to function as the virtual microcontroller 220. The FPGA and virtual microcontroller 220 will be referred to interchangeably herein. The base station 218 further includes a trace buffer 241, which stores trace path of the code. The pod, in certain embodiments, provides connection to the microcontroller 232 that permits external probing as well as interconnection with other circuitry as might be used to simulate a system under development.

Please replace paragraph beginning at page 11, line 22, which has been previously amended by a Preliminary Amendment, with the following amended paragraph:

The FPGA of the base station 218 of the current embodiment is designed to emulate the core processor functionality (microprocessor functions, Arithmetic Logic Unit function, RAM, and ROM memory functions) of the Cypress MicroSystems CY8C25xxx/26xxx series of microcontrollers. The CY8C25xxx/26xxx series of microcontrollers also incorporate limited I/O functions and an interrupt controller as well as programmable digital and analog circuitry. This circuitry need not be modeled using the FPGA 220. Instead the I/O read information, interrupt vector and other information can be passed to the

FPGA 220 from the ~~microprocessor~~ microcontroller 232 via interface ~~227~~ 226 as will be described later.

Please replace paragraph beginning at page 13, line 5, with the following amended paragraph:

In the embodiment described in connection with FIGURE 2, the actual production microcontroller 232 carries out its normal functions in the intended application and passes I/O information and other information needed for debugging to the base station 218 only at a ~~break-point~~ breakpoint or when in halt state. The virtual microcontroller 220 implemented within the FPGA of base station 218 serves to provide the operator with visibility into the core processor functions that are inaccessible in the production microcontroller 232. Thus, the FPGA 220, by virtue of operating in lock-step operation with the ~~microprocessor~~ microcontroller 232 provides an exact duplicate of internal CPU registers contents, memory contents, interrupt vectors and other useful debug information. Additionally, trace buffer 241 can be used to store information useful in trace operations that is gathered by the FPGA 220 during execution of the program under test. This architecture, therefore, permits the operator to have visibility into the inner workings of the microcontroller 232 without need to provide special boundouts and expensive circuitry on the microcontroller itself.

Please replace paragraph beginning at page 13, line 20, with the following amended paragraph:

The base station 218's FPGA based virtual microcontroller 220, operating under the control of host computer 210, carries out the core processor function of microcontroller 232 and thus contains a functionally exact copy ~~of the contents~~ of the CPU registers 238 contents and ~~memory~~ SRAM 237 contents of the production microcontroller 232. The ICE system starts both microcontrollers (production and virtual) at the same time and keeps them running in synchronization. One embodiment of the present invention provides a consistency check whenever the system is halted (i.e., when the system is not emulating), or when the system encounters a breakpoint (e.g., sleep, stall, internal start/stop, stalls, etc.).

Please replace paragraph beginning at page 14, line 3, with the following amended paragraph:

For example, whenever production microcontrollers 232 and virtual microcontroller 220 are running a code the execution of the code is in lock-step. In other words, the two microcontrollers are running the same code, they start running the code at the same time, they hit the same ~~break-point~~ breakpoint, and they stop at the same line of code. To ensure the integrity of the debugging process in production microcontrollers 232 and virtual microcontroller 220, a consistency check is performed at any opportune time. It is appreciated that computer system 210 and the host device 210 are ~~refereed~~ referred to interchangeably herein.

Please replace paragraph beginning at page 14, line 11, which has been previously amended by an Amendment mailed 1/10/2006 in response to an Office Action mailed 11/21/2005, with the following amended paragraph:

For example a consistency check may be conducted when the execution of the debugging operation is halted. The consistency check comprises: comparing a content of SRAM 225 and a content of SRAM 237, and comparing a content of CPU register 234 and a content of CPU register 238. The software in the host device 210 reads back the content of SRAM 225 and the content of SRAM 237 into memory 222. The software program residing in host device 210 compares the contents of the two SRAMs 225 and 237 to verify the consistency. If the contents of the two SRAMs 225 and 237 are not consistent the software in the host device 210 issues a signal indicating a "lock-step error". Similarly, the software in the host device 210 may compare a content of ~~ICE~~ CPU register 234 ~~238~~ and a content of production microcontroller CPU register 238 ~~234~~ for consistency verification. In a similar manner the software in host device 210 reads back a content of CPU register 238 and a content of CPU register 234 into memory 222. The software program residing in host device 210 compares the contents of the two CPU registers 234 and 238 and signals "lock-step error" if the contents of the two CPUs registers 234 and 238 are not matching.

Please replace paragraph beginning at page 14, line 26, which has been previously amended by an Amendment mailed 1/10/2006 in response to an Office Action mailed 11/21/2005, with the following amended paragraph:

When a lock-step error signal is detected, the user checks the trace buffers 241. The trace buffer 241, residing in the base station 218, keeps track of each line of code executed. Examining trace buffer 241 the user can determine which line of code caused the halt. Trace buffer also keeps track of the contents of CPU registers on each line of code. The user can back track the execution of each line of code and the associated CPU registers to find the exact line of code where the content of CPU register 238 and the content of CPU register 234 diverged as a result of a faulty code. Once the code is debugged the debugging process will resume. It is appreciated that CPU register 234 and SRAM 225 in the base station 218 and CPU register 238 and SRAM 237 in production microcontroller 232 are initialized with zeros to ensure the integrity of the tracing and the consistency checking.

Please replace paragraph beginning at page 15, line 11, which has been previously amended by an Amendment mailed 1/10/2006 in response to an Office Action mailed 11/21/2005, with the following amended paragraph:

Similarly, when a breakpoint ~~break-point~~ is encountered, a lock-step consistency check may be conducted. The lock-step operation of the microcontroller 232 and the virtual microcontroller 220 requires the virtual microcontroller 220 and production microcontroller 232 to start running a microcontroller code at the same time, run each line of the microcontroller code

at the same time, have the same CPU register content and the same SRAM content on both sides, and to stop at the exactly ~~the~~ same line of code when a breakpoint is encountered. Therefore, at a breakpoint the software in the host device 210 will conduct a consistency check as described above and if there is a mismatch of ~~memory~~ CPU register content or SRAM content the host device 210 will issue a "lock-step error" signal.